🛒 FreshCart – Multi-Seller Grocery Delivery Web App

📋 Project Overview

FreshCart is a comprehensive multi-seller grocery e-commerce platform built using the MERN stack (MongoDB, Express.js, React, Node.js). The application serves as a marketplace where multiple sellers can list their products and customers can browse, purchase, and track their orders. It integrates real-time inventory updates, secure payments, and cloud deployment.

🏗️ Architecture Analysis

Client-Server Architecture

- Frontend: React SPA with Vite build tool

- Backend: Express.js REST API with Socket.IO for real-time features

- Database: MongoDB with Mongoose ODM

- Deployment: Vercel (Frontend) + Render (Backend)

Key Architectural Patterns

- MVC Pattern: Clear separation of Models, Controllers, and Routes
- Context API: Centralized state management for React
- Middleware Pattern: Authentication and request processing
- Real-time Communication: Socket.IO for live updates

🔧 Technology Stack

Frontend Technologies

- React.js (Vite)
- TailwindCSS
- React Router DOM
- Axios
- Socket.IO Client
- Chart.js & React-ChartJS-2
- React Hot Toast
- jsPDF & html2canvas

Backend Technologies

- Node.js
- Express.js
- MongoDB + Mongoose
- Socket.IO
- JWT Authentication
- bcryptjs (Password Hashing)
- Stripe API
- Cloudinary (Image Storage)
- Multer (File Uploads)
- CORS

System Design

System Architecture Overview

FreshCart follows a 3-tier architecture integrated with real-time communication:

Frontend (Presentation Layer): React.js (Vite) + Tailwind CSS, JWT authentication, Axios API calls.

Backend (Application Layer): Node.js + Express.js, JWT-based authentication, Socket.IO, Stripe Checkout.

Database (Data Layer): MongoDB Atlas with Mongoose ORM for Users, Sellers, Products, Orders, and Addresses.

Modules and Components

- Frontend Modules: Home Page, Cart Module, Checkout, Orders Page, Seller Dashboard.
- Backend Modules: Authentication, Product, Order, and Socket.IO modules with Cloudinary and Stripe integration.

System Design Flow

- Customer Flow: Login, browse, add to cart, checkout (COD/Stripe), track order.
- Seller Flow: Register with Admin Key, add products, manage inventory, update order statuses.

External Integrations

- Cloudinary: Image storage.
- Stripe Checkout + Webhooks: Secure online payments with reliable webhook confirmation.
- Socket.IO: Real-time inventory synchronization.

Deployment Architecture

- Frontend: Vercel (CI/CD, environment variables).
- Backend: Render (Node.js + Express + Socket.IO, Stripe webhooks).
- Database: MongoDB Atlas (cloud-hosted, scalable).

Project Structure

Frontend Structure (/client):

```
├── src/
│   ├── components/ (UI components)
│   ├── pages/ (Customer + Seller Pages)
│   ├── context/ (Global State)
│   ├── utils/ (Utilities)
│   └── assets/ (Static files)
```

Backend Structure (/server):

```
├── controllers/ (Business logic)
├── models/ (Schemas: User, Seller, Product, Order, Address)
├── routes/ (API Endpoints)
├── middlewares/ (Authentication)
├── configs/ (Database, Cloudinary)
└── server.js (App entrypoint)
```

Core Features

- Dual Authentication: Separate JWT auth for customers and sellers.
- Multi-Seller Marketplace: Each seller manages their own products.
- Cart + Checkout System: COD and Stripe payments supported.
- Order Management: Full lifecycle tracking with status updates.
- Real-time Inventory: Stock updates via Socket.IO.
- Seller Dashboard: Manage products, view orders, track revenue.

Security Implementation

- JWT-based authentication with secure cookies.
- Password hashing with bcryptjs.
- Middleware route protection for sellers and customers.
- Database validation with Mongoose schemas.
- CORS configuration for controlled API access.

Payment Integration

- Stripe Checkout: Secure payment flow with webhook confirmation.
- Cash on Delivery: Alternative offline payment mode.

Deployment Links

Live App: https://fresh-cart-5sqh.vercel.app/

GitHub Repo: https://github.com/BhuvaneshAdithya45/FreshCart

Conclusion

FreshCart is a well-architected, feature-rich multi-seller grocery platform demonstrating modern full-stack web development practices. It integrates real-time inventory management, secure payments, scalable cloud deployment, and modular design. This project highlights skills in MERN stack, real-time systems, and production deployment.